Attorney's Docket No. RAL9-1999-0133/4269-97

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re: Fluke et al.
Serial No.: 09/607,074
Filed: June 29, 2000
For: METHODS, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR DEFERRED COMPUTER PROGRAM TRACING

Confirmation No.: 7844
Group Art Unit: 2122
Examiner: E. Kiss

Date: February 6, 2004

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPELLANTS' BRIEF ON APPEAL UNDER 37 C.F.R. §1.192

Sir:

This Appeal Brief is filed pursuant to the "Notice of Appeal to the Board of Patent Appeals and Interferences" filed December 9, 2003.

Real Party In Interest

The real party in interest is assignee International Business Machines, Inc., Armonk, New York.

Related Appeals and Interferences

Appellants are aware of no appeals or interferences that would be affected by the present appeal.

Status of Claims

Appellants appeal the final rejection of Claims 1 - 57, which as of the filing date of this Brief remain under consideration. The claims at issue as included in Appellants' response to the final Office Action of July 16, 2003 are attached hereto as Appendix A.

Status of Amendments

Two responses have been filed in the present case: An "Amendment" was filed April 21, 2003 in response to an Office Action mailed January 21, 2003 (hereinafter "Office Action"). A "Response After Final" was filed September 15, 2003 in response to a final Office Action mailed July 16, 2003 (hereinafter "Final Action"). An Advisory Action was mailed October 22, 2003. No claims have been canceled in prosecuting the present application; therefore, Claims 1 - 57 remain for consideration on the present appeal.

Summary of the Invention

According to some embodiments of the present invention, an application (*e.g.*, a computer program) prints data by invoking a print function with a format argument and, optionally, at least one data argument. (Specification, page 10, line 3 - page 11, line 4). The format argument is a pointer to a memory location in an address space of the application. (Specification, page 11, lines 11 - 12). The format argument and any data arguments are saved in a deferred trace data buffer. (Specification, page 11, lines 5 - 10). The print function returns to the application then, sometime after the print function has returned, the deferred trace data buffer is processed and the format argument and/or any data arguments are printed. (Specification, page 12, lines 18 - 26). By saving the format argument and any data argument(s) to a memory buffer instead of parsing and formatting the arguments in real-time, program efficiency may be improved and the impact of the print function on a software test scenario may be reduced. In other embodiments of the present invention, the deferred trace data buffer and the memory contents comprising the address space of the application are saved in a non-volatile storage medium. (Specification, page 13, lines 20 - 23).

Issues

I. Are Claims 1, 9, 20, 28, 39, and 47 properly rejected under 35 U.S.C. §103(a) as being as being unpatentable over U. S. Patent No. 5,983,366 to King (hereinafter "King") in view of the document "The Visual C++ Debugging Environment" authored by Keith Bugg (hereinafter "Bugg")?

II. Are Claims 7, 26, and 45 properly rejected under 35 U.S.C. §103(a) as being as being unpatentable over King in view of Bugg?

Grouping of Claims

For purposes of this appeal, Claims 1 - 6, 8 - 25, 27 - 44, and 46 - 57 (Group I) may be considered as standing or falling together, and Claims 7, 26, and 45 (Group II) may be considered as standing or falling together. Appellants respectfully submit that the Group II Claims are separately patentable from the Group I Claims because the Group II Claims recite "saving...a memory contents comprising the address space of the application in a non-volatile medium," which provides separate grounds for patentability.

Argument

I. Introduction to 35 U.S.C. §103 Analysis

A determination under §103 that an invention would have been obvious to someone of ordinary skill in the art is a conclusion of law based on fact. *Panduit Corp. v. Dennison Mfg. Co.* 810 F.2d 1593, 1 U.S.P.Q.2d 1593 (Fed. Cir. 1987), *cert. denied*, 107 S.Ct. 2187. After the involved facts are determined, the decision maker must then make the legal determination of whether the claimed invention as a whole would have been obvious to a person having ordinary skill in the art at the time the invention was unknown, and just before it was made. *Id.* at 1596. The United States Patent and Trademark Office (USPTO) has the initial burden under §103 to establish a *prima facie* case of obviousness. *In re Fine*, 837 F.2d 1071, 5 U.S.P.Q.2d 1596, 1598 (Fed. Cir. 1988).

To establish a prima facie case of obviousness, the prior art reference or references when combined must teach or suggest *all* the recitations of the claims, and

there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. M.P.E.P. §2143. The mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination. M.P.E.P. §2143.01, citing *In re Mills*, 916 F.2d 680, 16 U.S.P.Q.2d 1430 (Fed. Cir. 1990). As recently emphasized by the Court of Appeals for the Federal Circuit, to support combining references, evidence of a suggestion, teaching, or motivation to combine must be **clear and particular**, and this requirement for clear and particular evidence is not met by broad and conclusory statements about the teachings of references. *In re Dembiczak*, 50 U.S.P.Q.2d 1614, 1617 (Fed. Cir. 1999). In an even more recent decision, the Court of Appeals for the Federal Circuit has stated that, to support combining or modifying references, there must be **particular** evidence from the prior art as to the reason the skilled artisan, with no knowledge of the claimed invention, would have selected these components for combination in the manner claimed. *In re Kotzab*, 55 U.S.P.Q.2d 1313, 1317 (Fed. Cir. 2000).

Appellants respectfully submit that the pending claims are patentable over the cited references for at least the reason that the cited references do not disclose or suggest, among other things, invoking a print function with a format argument and saving the format argument in a deferred trace data buffer where the format argument is a pointer to a memory location in an address space of the application. The patentability of the pending claims is discussed in detail hereinafter.

A. Independent Claims 1, 9, 20, 28, 39, and 47 are Patentable

Independent Claims 1, 9, 20, 28, 39, and 47 stand rejected under 35 U.S.C. §103(a) as being unpatentable over King in view of Bugg.

Independent Claims 1, 9, 20, 28, 39, and 47 are directed to methods, systems, and computer program products for printing data from an application in which a print function is invoked with a format argument and the format argument is saved in a deferred trace data buffer. The format argument is a pointer to a memory location in an address space of the application. Embodiments including this aspect of the present invention are described, for example, at page 11, lines 5 - 16 of the Specification.

The Final Action acknowledges that King does not disclose "the format argument being a pointer to a memory location in an address space of the application." (Final Action, page 5). The Final Action does assert, however, that Bugg teaches "a format argument a debugging information output command being a pointer to a memory location in an address space of an application..." (Final Action, page 5). Furthermore, in rejecting Claim 10, the Final Action again acknowledges that King does not disclose "the format argument being a pointer to a memory location in an address space of the application, and saving the pointer in the deferred trace data buffer." (Final Action, page 11). The Final Action does assert, however, that U. S. Patent No. 6,282,701 to Wygodny et al. (hereinafter "Wygodny") teaches "displaying a pointer (for example, variable names) and the contents of the memory referred to by the pointer as part of a trace output display..." (Final Action, page 11).

Appellants respectfully submit that the King, Wygodny, and Bugg references contain no description therein to suggest to or motivate one skilled in the art to modify King's computer program tracing system with the teachings of either Wygodny or Bugg. In fact, Appellants respectfully submit that the disclosures of King, Wygodny, and Bugg teach against such a combination as the resulting computer program tracing system as alleged by the Final Action would be inoperable.

King explains at column 19, lines 17 - 27, that a computer program may perform a trace by calling a trace macro, which in turns calls a trace function that includes a numerical identification of the trace message and a pair of parameters. In King's example, the trace message is identified as number 7292. The parameters and trace identification are packed into a message and transmitted from the data processing system 252 to the host processor 254. (King, col. 19, lines 40 - 44). Once the message is received at the host processor 254, the host processor 254 looks "up the trace id and then correctly unpack the trace message and display the trace string and its parameters in the way defined in the trace control file..." (King, col. 19, lines 50 - 52). Because King describes processing the trace message, which includes a trace identification and parameters, on a different processor (host processor 254) than the processor executing the computer program that is being traced (data processing system 252), replacing the trace identification with a pointer would not work because the host processor 254 does not have access to the address space of the data

processing system 252. That is, a pointer to a memory location in the data processing system 252 is useless to the host processor 254.

In response to the foregoing analysis, the Final Action states that the "format argument pointers described in Bugg are inherently de-referenced prior to being output/stored..." (Final Action, page 3). The Final Action further states that "in the online debugger system of Wygodny, the host processor does have access to the client application address space.." (Final Action, page 3). For these reasons, the Final Action maintains that modifying the King reference with the teachings of Bugg or Wygodny would not render the King system inoperable. (Final Action, page 3). Appellants respectfully disagree with this interpretation of the King, Bugg, and Wygodny references.

As discussed above, because King describes processing the trace message on a different processor than the processor executing the computer program that is being traced and because the two processors do not have access to the same address space, King's computer program tracing system would not work if a pointer is used instead of the trace identification. This is because the pointer would be based on the address space of the data processing system 252 rather than the host processor 254. Appellants further note that the claim language describes the format argument as being a pointer. Thus, it is irrelevant whether Bugg inherently teaches de-referencing a pointer first as the claim language does not recite the contents of a memory location pointed to by a pointer, but simply a pointer.

In response to the assertion in the Final Action that Wygodny discloses a system in which the host processor has access to the client application address space, Appellants respectfully disagree. As shown in FIG. 2 of Wygodny, the analyzer 106 and the client 102 share a trace buffer 105, but the analyzer does not have access to the address space of the user application executing on the client 102.

Accordingly, Appellants respectfully submit that one skilled in the art would not be motivated to replace the trace identification described in King with a pointer as described in Wygodny or Bugg as such a replacement would render King's computer program tracing system inoperable.

For at least the foregoing reasons, Appellants respectfully submit that independent Claims 1, 9, 20, 28, 39, and 47 are patentable over the cited references

and that dependent Claims 2 - 8, 10 - 19, 21 - 27, 29 - 38, 40 - 46, and 48 - 57 are patentable at least by virtue of their depending from an allowable claim. Accordingly, Appellants respectfully request that the rejection of Claims 1 - 57 (Groups I and II) be reversed based on the failure of the Examiner to establish a prima facie case of obviousness under 35 U.S.C. §103 for at least these reasons.

B. Claims 7, 26, and 45 are Patentable

Dependent Claims 7, 26, and 45 (Group II) stand rejected under 35 U.S.C. §103(a) as being as being unpatentable over King in view of Bugg. With regard to Claims 7, 26, and 45, these claims include all of the recitations from independent Claims 1, 20, and 39, respectively, and are, therefore, patentable over the cited references for at least the reasons stated above. In addition, Appellants submit that these claims are separately patentable as none of the cited references described or suggest "saving...a memory contents comprising the address space of the application in a non-volatile medium."

The Final Action alleges that Bugg teaches "sending debugging output, including format and data arguments to a file." (Final Action, page 3). Appellants respectfully submit that Bugg describes the ability to send a processed debug report, including a file name, line number, and a formatted message, to a file, debugger, or message window. The Final Action asserts that the filename, line number, and module name are indicators of the address space of the application. (Final Action, page 3). Appellants submit that whether the filename, line number, and module name may provide an indication of the address space or not is irrelevant. The filename, line number, and module name are not the actual address space of the application, which is a defined portion of memory where the application executes. Appellants submit that Bugg contains no description of saving the address space of the application in which the _CrtDbgReport() function is called to a non-volatile medium.

Therefore, Appellants submit that, in addition to the foregoing reasons, Claims 7, 26, and 45 are separately patentable for at least these additional reasons. Accordingly, Appellants respectfully request that the rejection of Claims 7, 26, and 45 (Group II) be reversed based on the failure of the Examiner to establish a prima facie case of obviousness under 35 U.S.C. §103 for at least these additional reasons.

II. Conclusion

In summary, Appellants respectfully submit that, with respect to Claims 1 - 57 (Groups I and II) the cited references do not teach all of the recitations of the claims. Accordingly, Appellants respectfully request reversal of the rejection of Claims 1 - 57 (Groups I and II) based on the cited references.

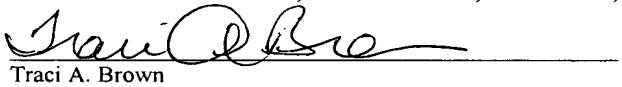
Respectfully submitted,

D. Scott Moore
Registration No. 42,011

Myers Bigel Sibley & Sajovec, P.A.
P. O. Box 37428
Raleigh, North Carolina 27627
Telephone: (919) 854-1400
Facsimile: (919) 854-1401
Customer No. 20792

Certificate of Mailing under 37 CFR 1.8

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on February 6, 2004.



Traci A. Brown

APPENDIX A

1. A method of printing data from an application, comprising the steps of:

invoking a print function with a format argument that is a pointer to a memory location in an address space of the application and at least one data argument from the application;

saving the format argument and the at least one data argument in a deferred trace data buffer;

returning to the application that invoked the print function; then

processing the deferred trace data buffer to print the at least one data argument.

2. A method as recited in Claim 1, wherein the step of processing the deferred trace data buffer to print the at least one data argument comprises the steps of:

retrieving the format argument and the at least one data argument from the deferred trace data buffer;

formatting the at least one data argument based on the format argument; and

printing the formatted at least one data argument.

3. A method as recited in Claim 2, wherein the step of formatting the at least one data argument based on the format argument comprises the steps of:

determining if the format argument specifies a character string conversion; and

printing an address of a respective one of the at least one data argument that corresponds to the character string conversion.

4. A method as recited in Claim 1, further comprising the step of:

determining if a deferred print flag has been set.

5. A method as recited in Claim 4, wherein the step of saving the format argument and the at least one data argument in the deferred trace data buffer comprises the step of:

saving the at least one data argument in the deferred trace data buffer if the deferred print flag has been set; and

wherein the step of processing the deferred trace data buffer to print the at least one data argument comprises the step of:

processing the deferred trace data buffer to print the at least one data argument if the deferred print flag has been set.

6. A method as recited in Claim 1, wherein the step of saving the format argument and the at least one data argument in the deferred trace data buffer and the step of processing the deferred trace data buffer to print the at least one data argument are performed in different execution threads.

7. A method as recited in Claim 1, further comprising the step of:

saving the deferred trace data buffer and a memory contents comprising the address space of the application in a non-volatile storage medium.

8. A method as recited in Claim 7, wherein the step of saving the format argument and the at least one data argument in the deferred trace data buffer is performed on a first computing machine and the step of processing the deferred trace data buffer to print the at least one data argument is performed on a second computing machine, the second computing machine being different from the first computing machine and having access to the address space of the application via the non-volatile storage medium.

9. A method of printing data from an application, comprising the steps of:

invoking a print function with a format argument that is a pointer to a memory location in an address space of the application from the application;

saving the format argument in a deferred trace data buffer;

returning to the application that invoked the print function; then

processing the deferred trace data buffer to print the format argument.

10. A method as recited in Claim 9, wherein the step of saving the format argument in the deferred trace data buffer comprises the step of:

saving the pointer in the deferred trace data buffer

11. A method as recited in Claim 10, wherein the step of processing the deferred trace data buffer to print the format argument comprises the step of:

processing the deferred trace data buffer to print a contents of the memory location in the address space of the application that is referenced by the pointer.

12. A method as recited in Claim 11, wherein the step of saving the pointer in the deferred trace data buffer and the step of processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer are performed in different execution threads.

13. A method as recited in Claim 11, further comprising the step of:

saving the deferred trace data buffer and a memory contents comprising the address space of the application in a non-volatile storage medium.

14. A method as recited in Claim 13, wherein the step of saving the pointer in the deferred trace data buffer is performed on a first computing machine and the step of processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer is performed on a second computing machine, the second computing machine being different from the first computing machine and having access to the deferred trace data buffer and the address space of the application via the non-volatile storage medium.

15. A method as recited in Claim 9, wherein the step of saving the format argument in the deferred trace data buffer comprises the step of:

saving a contents of the memory location in the address space of the application that is referenced by the pointer in the deferred trace data buffer.

16. A method as recited in Claim 15, wherein the step of processing the deferred trace data buffer to print the format argument comprises the step of:

processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer.

17. A method as recited in Claim 16, wherein the step of saving the contents of the memory location in the address space that is referenced by the pointer in the deferred trace data buffer and the step of processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer are performed in different execution threads.

18. A method as recited in Claim 16, further comprising the step of:

saving the deferred trace data buffer to a non-volatile storage medium.

19. A method as recited in Claim 18, wherein the step of saving the contents of the memory location in the e address space that is referenced by the pointer in the deferred trace data buffer is performed on a first computing machine and the step of processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer is performed on a second computing machine, the second computing machine being different from the first computing machine and having access to the deferred trace data buffer via the non-volatile storage medium.

20. A system for printing data from an application, comprising:

means for invoking a print function with a format argument that is a pointer to a memory location in an address space of the application and at least one data argument from the application;

means for saving the format argument and the at least one data argument in a deferred trace data buffer;

means for returning to the application that invoked the print function; and

means for processing the deferred trace data buffer to print the at least one data argument after returning to the application that invoked the print function.

21. A system as recited in Claim 20, wherein the means for processing the deferred trace data buffer to print the at least one data argument comprises:

means for retrieving the format argument and the at least one data argument from the deferred trace data buffer;

means for formatting the at least one data argument based on the format argument; and

means for printing the formatted at least one data argument.

22. A system as recited in Claim 21, wherein the means for formatting the at least one data argument based on the format argument comprises:

means for determining if the format argument specifies a character string conversion; and

means for printing an address of a respective one of the at least one data argument that corresponds to the character string conversion.

23. A system as recited in Claim 20, further comprising:

means for determining if a deferred print flag has been set.

24. A system as recited in Claim 23, wherein the means for saving the format argument and the at least one data argument in the deferred trace data buffer comprises:

means for saving the at least one data argument in the deferred trace data buffer if the deferred print flag has been set; and

wherein the means for processing the deferred trace data buffer to print the at least one data argument comprises:

means for processing the deferred trace data buffer to print the at least one data argument if the deferred print flag has been set.

25. A system as recited in Claim 20, wherein the means for saving the format argument and the at least one data argument in the deferred trace data buffer and the means for processing the deferred trace data buffer to print the at least one data argument execute in different execution threads.

26. A system as recited in Claim 20, further comprising:

means for saving the deferred trace data buffer and a memory contents comprising the address space of the application in a non-volatile storage medium.

27. A system as recited in Claim 26, wherein the means for saving the format argument and the at least one data argument in the deferred trace data buffer executes on a first computing machine and the means for processing the deferred trace data buffer to print the at least one data argument executes on a second computing machine, the second computing machine being different from the first computing machine and having access to the address space of the application via the non-volatile storage medium.

28. A system for printing data from an application, comprising:

means for invoking a print function with a format argument that is a pointer to a memory location in an address space of the application from the application;

means for saving the format argument in a deferred trace data buffer;

means for returning to the application that invoked the print function; and

means for processing the deferred trace data buffer to print the format argument after returning to the application that invoked the print function.

29. A system as recited in Claim 28, wherein the means for saving the format argument in the deferred trace data buffer comprises:

means for saving the pointer in the deferred trace data buffer.

30. A system as recited in Claim 29, wherein the means for processing the deferred trace data buffer to print the format argument comprises:

means for processing the deferred trace data buffer to print a contents of the memory location in the address space of the application that is referenced by the pointer.

31. A system as recited in Claim 30, wherein the means for saving the pointer in the deferred trace data buffer and the means for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer execute in different execution threads.

32. A system as recited in Claim 30, further comprising:

means for saving the deferred trace data buffer and a memory contents comprising the address space of the application in a non-volatile storage medium.

33. A system as recited in Claim 32, wherein the means for saving the pointer in the deferred trace data buffer executes on a first computing machine and the means for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer executes on a second computing machine, the second computing machine being different from the first computing machine and having access to the deferred trace data buffer and the address space of the application via the non-volatile storage medium.

34. A system as recited in Claim 28, wherein the means for saving the format argument in the deferred trace data buffer comprises:

means for saving a contents of the memory location in the address space of the application that is referenced by the pointer in the deferred trace data buffer.

35. A system as recited in Claim 34, wherein the means for processing the deferred trace data buffer to print the format argument comprises:

means for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer.

36. A system as recited in Claim 35, wherein the means for saving the contents of the memory location in the address space that is referenced by the pointer in the deferred trace data buffer and the means for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer execute in different execution threads.

37. A system as recited in Claim 35, further comprising:

means for saving the deferred trace data buffer to a non-volatile storage medium.

38. A system as recited in Claim 37, wherein the means for saving the contents of the memory location in the address space that is referenced by the pointer in the deferred trace data buffer executes on a first computing machine and the means for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer executes on a second computing machine, the second computing machine being different from the first computing machine and having access to the deferred trace data buffer via the non-volatile storage medium.

39. A computer program product for printing data from an application, comprising:

a computer readable storage medium having computer readable program code embodied therein, the computer readable program code comprising:

computer readable program code for invoking a print function with a format argument that is a pointer to a memory location in an address space of the application and at least one data argument from the application;

computer readable program code for saving the format argument and the at least one data argument in a deferred trace data buffer;

computer readable program code for returning to the application that invoked the print function; and

computer readable program code for processing the deferred trace data buffer to print the at least one data argument after returning to the application that invoked the print function.

40. A computer program product as recited in Claim 39, wherein the computer readable program code for processing the deferred trace data buffer to print the at least one data argument comprises:

computer readable program code for retrieving the format argument and the at least one data argument from the deferred trace data buffer;

computer readable program code for formatting the at least one data argument based on the format argument; and

computer readable program code for printing the formatted at least one data argument.

41. A computer program product as recited in Claim 40, wherein the computer readable program code for formatting the at least one data argument based on the format argument comprises:

computer readable program code for determining if the format argument specifies a character string conversion; and

computer readable program code for printing an address of a respective one of the at least one data argument that corresponds to the character string conversion.

42. A computer program product as recited in Claim 39, further comprising:

computer readable program code for determining if a deferred print flag has been set.

43. A computer program product as recited in Claim 42, wherein the computer readable program code for saving the format argument and the at least one data argument in the deferred trace data buffer comprises:

computer readable program code for saving the at least one data argument in the deferred trace data buffer if the deferred print flag has been set; and

wherein the computer readable program code for processing the deferred trace data buffer to print the at least one data argument comprises:

computer readable program code for processing the deferred trace data buffer to print the at least one data argument if the deferred print flag has been set.

44. A computer program product as recited in Claim 39, wherein the computer readable program code for saving the format argument and the at least one data argument in the deferred trace data buffer and the computer readable program code for processing the deferred trace data buffer to print the at least one data argument execute in different execution threads.

45. A computer program product as recited in Claim 39, further comprising:

computer readable program code for saving the deferred trace data buffer and a memory contents comprising the address space of the application in a non-volatile storage medium.

46. A computer program product as recited in Claim 45, wherein the computer readable program code for saving the format argument and the at least one data argument in the deferred trace data buffer executes on a first computing machine and the computer readable program code for processing the deferred trace data buffer to print the at least one data argument executes on a second computing machine, the second computing machine being different from the first computing machine and having access to the address space of the application via the non-volatile storage medium.

47. A computer program product for printing data from an application, comprising:

a computer readable storage medium having computer readable program code embodied therein, the computer readable program code comprising:

computer readable program code for invoking a print function with a format argument that is a pointer to a memory location in an address space of the application from the application;

computer readable program code for saving the format argument in a deferred trace data buffer;

computer readable program code for returning to the application that invoked the print function; and

computer readable program code for processing the deferred trace data buffer to print the format argument after returning to the application that invoked the print function.

48. A computer program product as recited in Claim 47, wherein the computer readable program code for saving the format argument in the deferred trace data buffer comprises:

computer readable program code for saving the pointer in the deferred trace data buffer.

49. A computer program product as recited in Claim 48, wherein the computer readable program code for processing the deferred trace data buffer to print the format argument comprises:

computer readable program code for processing the deferred trace data buffer to print a contents of the memory location in the address space of the application that is referenced by the pointer.

50. A computer program product as recited in Claim 49, wherein the computer readable program code for saving the pointer in the deferred trace data buffer and the computer readable program code for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer execute in different execution threads.

51. A computer program product as recited in Claim 49, further comprising:

computer readable program code for saving the deferred trace data buffer and a memory contents comprising the address space of the application in a non-volatile storage medium.

52. A computer program product as recited in Claim 51, wherein the computer readable program code for saving the pointer in the deferred trace data buffer executes on a first computing machine and the computer readable program code for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer executes on a second computing machine, the second computing machine being different from the first computing machine and having access to the deferred trace data buffer and the address space of the application via the non-volatile storage medium.

53. A computer program product as recited in Claim 47, wherein the computer readable program code for saving the format argument in the deferred trace data buffer comprises:

computer readable program code for saving a contents of the memory location in the address space of the application that is referenced by the pointer in the deferred trace data buffer.

54. A computer program product as recited in Claim 53, wherein the computer readable program code for processing the deferred trace data buffer to print the format argument comprises:

computer readable program code for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer.

55. A computer program product as recited in Claim 54, wherein the computer readable program code for saving the contents of the memory location in the address space that is referenced by the pointer in the deferred trace data buffer and the computer readable program code for processing the deferred trace data buffer to

print the contents of the memory location in the address space of the application that is referenced by the pointer execute in different execution threads.

56. A computer program product as recited in Claim 54, further comprising:

computer readable program code for saving the deferred trace data buffer to a non-volatile storage medium.

57. A computer program product as recited in Claim 56, wherein the computer readable program code for saving the contents of the memory location in the address space that is referenced by the pointer in the deferred trace data buffer executes on a first computing machine and the computer readable program code for processing the deferred trace data buffer to print the contents of the memory location in the address space of the application that is referenced by the pointer executes on a second computing machine, the second computing machine being different from the first computing machine and having access to the deferred trace data buffer via the non-volatile storage medium.